ISSN 2337-6805

PENGAMANAN DATA DIGITAL SIGNATURE DENGAN MENGGUNAKAN METODE ALGORITMA RSA DAN HASH

SECURING DIGITAL SIGNATURE DATA USING THE RSA AND HASH ALGORITHM METHODS

Yansiska[1], Aryo Nur Utomo [2]
Program Studi Teknik Informatika, Fakultas Sains dan Teknologi Informasi
Institut Sains dan Teknologi Nasional
Jl. Moh. Kahfi II, Bhumi Srengseng Indah, Jakarta Selatan 12640
[1]yansiska01@gmail.com, [2]aryo.nurutomo@gmail.com

ABSTRAKSI
Penelitian ini bertujuan untuk menjaga keaslian pesan yang akan dikirim untuk memberikan jaminan kepada penerima bahwa pesan tersebut bebas dari perubahan atau modifikasi yang dilakukan oleh pihak lain. Jika terjadi suatu perubahan terhadap pesan, maka penerima akan mengetahui bahwa pesan tersebut sudah tidak lagi terjaga keasliannya, sehingga penerima pesan terhindar dari penggunaan data yang salah. Untuk menjaga keaslian data digunakan teknik digital signature dengan menggunakan algoritma RSA sebagai algoritma kunci publik dan fungsi hash untuk menghasilkan message digest dari pesan yang dikirim. Kombinasi dari kedua algoritma tersebut akan menghasilkan digital signature dari setiap file atau dokumen yang dapat dijaga keasliannya.
Kata Kunci : Digital Signature, Algoritma RSA, Message Digest, Fungsi Hash.

ABSTRACT
This research aims to maintain the authenticity of the message to be sent to provide assurance to the recipient that the message is free from changes or modifications made by other parties. If there is a change to the message, the recipient will know that the original message is no longer maintained, so that the recipient of the message avoids using the wrong data. To maintain data authenticity, digital signature techniques are used using the RSA algorithm as a public key algorithm and a hash function to produce a message digest of the messages sent. The combination of the two algorithms will produce a digital signature for each file or document that can be maintained as authentic.
Keywords : *Digital Signature, RSA Algorithm, Message Digest, Hash Function.*

1. PENDAHULUAN

Perkembangan teknologi yang semakin pesat tidak dapat dipungkiri telah mengubah cara kerja berbagai kegiatan dalam bidang kehidupan manusia mulai dari perusahaan sampai pemerintah. Dengan perkembangan teknologi saat ini pertukaran informasi antar pihak sangat diperlukan. Jika keamanan pertukaran informasi tidak bisa di jaga, maka pihak lain dapat memanfaatkan informasi tersebut sehingga akan merugikan pihak-pihak yang berhak atas informasi tersebut.

Ada beberapa bentuk ancaman terhadap pertukaran informasi seperti penyadapan, pencurian dan pemalsuan informasi. Untuk itu keamanan dari pertukaran informasi tersebut sangatlah diperlukan. Kriptografi adalah salah satu solusi yang tepat untuk menjaga kerahasiaan dan keaslian data serta dapat meningkatkan keamanan suatu data.

Tanda tangan digital digunakan untuk menjamin integritas data, otentikasi dan nirpenyangkalan (tidak dapat disangkal). Tanda tangan digital merupakan salah satu teknik yang dapat digunakan untuk menjaga keaslian data, sehingga penerima mendapatkan jaminan untuk mengetahui bahwa data yang diterima merupakan data asli atau data palsu.

Menurut (Ihwani, 2016) kriptografi Digital Signature Algorithm (DSA) juga membutuhkan fungsi hash, fungsi hash merupakan fungsi yang menerima masukan string yang panjangnya sembarang dan mengkonversinya menjadi string keluaran yang panjangnya tetap (fixed), umumnya berukuran jauh lebih kecil daripada ukuran string semula, hasil konversi pesan tersebut akan disamakan dengan hasil dekripsi dari proses kriptografi DSA untuk otentikasi dan integritas dari keaslian pesan.

Pada penelitian sebelumnya yang dilakukan oleh (Putri & Manullang, 2018) dengan mengimplementasikan kriptogafi digital signature menggunakan secure hash algorithm (SHA-1) menyatakan bahwa ketika melakukan proses hashing pada file yang berukuran besar (200kb), maka proses pencarian nilai SHA-1 akan menjadi lambat. Pada proses digital signature sendiri memerlukan suatu algoritma asimetris untuk melakukan pertukaran data agar proses

ISSN 2337-6805

pengiriman dan penerimaan lebih terjaga integritas keamananya. Oleh karena itu, diperlukan algoritma yang tepat untuk proses pembuatan pengamanan data digital signature.
RSA merupakan algoritma kriptografi asimetris, dimana kunci yang digunakan untuk mengenkripsi berbeda dengan yang digunakan untuk mendekripsi. Hash adalah algoritma yang dipakai untuk mengubah informasi. Data yang dimasukkan akan diolah menjadi angka, huruf, atau karakter lain menjadi karakter terenkripsi tanpa mengubah ukuran. Data yang terenkripsi dengan menggunakan fungsi hash tidak dapat dikembalikan lagi.

Hal ini yang membuat algoritma tersebut dikenal sebagai One Way Function atau encryption satu arah. Fungsi hash dari md5 merupakan salah satu algoritma yang dapat menerima input dengan panjang sembarang dan akan menghasilkan output berupa message digest dengan panjang 128 bit dimana akan mempercepat proses pencarian file. SHA-1 (Secure Hash Algorithm) merupakan fungsi hash satu arah. SHA-1 menerima masukan berupa pesan dengan menghasilkan message digest (MD) dengan panjang 160 bit yang memungkinkan dapat memperlambat proses pencarian file yang berukuran besar.

Oleh karena itu, dari penjelasan yang telah disampaikan dalam hal ini bagaimana jika algoritma RSA dan juga Hash di aplikasikan kedalam bentuk pengamanan data digital signature.

2. METODOLOGI PENELITIAN

Objek Penelitian

Objek dari penelitian ini adalah masalah keamanan dan keaslian tanda tangan digital. Keamanan dan keaslian tanda tangan digital ini terletak pada pengiriman pesan kepada penerima yang termasuk didalamnya pembuktian bahwa pesan yang dikirim berasal dari orang yang seharusnya, ataupun pesan yang dikirim memang tidak pernah mengalami perubahan sama sekali setelah ditandatangani oleh pembuat pesan tersebut. Sehingga dalam hal ini diperlukan suatu cara untuk meminimalisir kemungkinan terjadinya hal yang tidak di inginkan.

Penggunaan algoritma *RSA* dan juga fungsi *hash* adalah salah satu cara untuk menanggulangi masalah terhadap pengiriman pesan tersebut. Algoritma yang digunakan yaitu *RSA* dan juga fungsi *hash*, keduanya di pilih karena relatif sederhana untuk di implementasikan dan juga cukup kuat, karena keamanan dengan algoritma *RSA* terletak pada sulitnya untuk memfaktorkan bilangan yang besar menjadi faktor-faktor prima dan juga fungsi *hash* merupakan salah satu teknik penyandian pesan yang sangat bisa diandalkan, dengan kekuatannya terletak pada pengubahan satu karakter di dalam dokumen atau data yang dipetakan akan menyebabkan perubahan yang cukup signifikan pada hasil fungsi *hash*-nya, sehingga dapat dengan mudah diketahui bahwa data atau dokumennya telah berubah.

Aplikasi *digital signature* ini berfungsi untuk memastikan keamanan dan juga keaslian dari pesan yang dikirim. Tanda tangan digital didapatkan dari *hash* terhadap pesan, nilai *hash* merupakan ringkasan kode dari pesan.
Aplikasi *digital signature* ini menggunakan basis data, untuk menyimpan direktori dari *public key*. Hasil *signature* (*key* yang didapat setelah di *generate* dan file *hash*) yang dibuat juga dapat disimpan ke penyimpanan lokal pada perangkat yang menjalankan aplikasi ini dengan ekstensi untuk *public key* yaitu *.publickey*, untuk *private key* yaitu *.privatekey* dan untuk file *hash* yaitu *.hashencr*. Untuk file pesan yang dikirim dan diterima juga merupakan file dengan ekstensi .txt.

Instrumen Penelitian

Instrumen penelitian adalah alat yang dipakai dalam sebuah kegiatan penelitian yang khususnya sebagai pengukuran dan pengumpulan data.

Perangkat Keras (*Hardware*)

Perangkat komputer yang digunakan dalam penelitian untuk melakukan pembuatan aplikasi ini memiliki spesifikasi sebagai berikut :

1. Laptop ASUS TUF GAMING FX504
2. Processor Intel Core i7-8750H 2.2 GHz
3. RAM 16 GB DDR4
4. Hardisk: 1TB SSHD + SSD 516GB
5. Monitor 15.6 inch
6. OS Windows 11

Perangkat Lunak (*Software*)

Perangkat lunak yang digunakan dalam penelitian untuk pembuatan aplikasi adalah sebagai berikut :

1. *Windows* 11 64-*bit*
2. *Java Runtime Environment* (*JRE*)
3. *Java Development Kit* (*JDK*)
4. *Apache NetBeans IDE* 16
5. *XAMPP*

Metode Penelitian

Metode penelitian adalah strategi, proses, atau teknik yang digunakan oleh peneliti dalam upaya pengumpulan data atau bukti agar selanjutnya dapat dilakukan dianalisis guna

ISSN 2337-6805

mengungkap informasi baru atau menciptakan pemahaman yang lebih baik tentang suatu topik atau tema permasalahan.

Metode Pengumpulan Data

Metode pengumpulan data yang digunakan dalam penelitian ini antara lain yaitu dengan menggunakan metode literatur, yaitu dengan melakukan pengumpulan data dan referensi dari berbagai jenis buku, *website*, *e-book*, serta jurnal acuan yang berkaitan dengan penelitian dan perangkat yang digunakan.

Metode Pengolahan Data

Metode pengolahan data yang digunakan dalam penelitian ini antara lain:

1. Fase Perencanaan (*Planning*)
2. Rencana program aplikasi
3. Analisa kebutuhan aplikasi
4. Fase Pengembangan (*Development*)
5. Pembuatan Kode (*Coding*)
6. Pengujian program aplikasi (*Testing*)
7. Fase Pemeliharaan (*Maintenance*)
8. Koreksi program aplikasi (*Corection*)
9. Adaptasi program aplikasi (*Adaptive*)

Metode Pembuatan Aplikasi

Metode yang digunakan dalam pembuatan aplikasi *digital signature* yaitu *extreme programming*. Untuk analisis dan desain yang digunakan adalah *Unified Modeling Language* (*UML*), sedangkan metode yang digunakan untuk *testing* atau pengujian aplikasi adalah *black box*.

Ruang Lingkup Sistem

Adapun ruang lingkup pada aplikasi *Digital Signature* dengan menggunakan Algoritma *RSA* dan *Hash* adalah sebagai berikut:

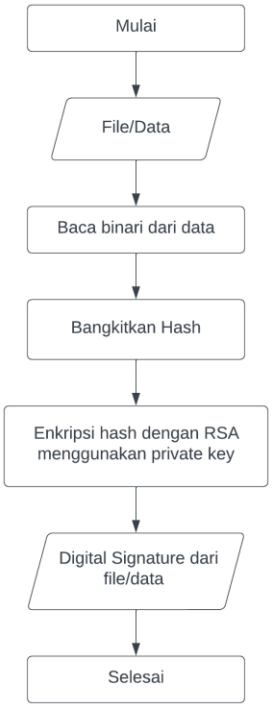
1. Aplikasi ini merupakan application dekstop yang dapat dijalankan pada *PC* atau *Laptop* menggunakan sistem operasi *windows*.
2. Dalam menggunakan aplikasi ini, sistem operasi *windows* yang digunakan *windows* 7 ke atas.
3. Aplikasi ini digunakan untuk mengakomodasi keamanan dalam pertukaran dokumen digital dengan menggunakan algoritma *RSA* dan juga *Hash*.

Gambaran Umum Sistem

Dalam sistem pembuatan aplikasi *Digital Signature* dengan menggunakan algoritma *RSA* dan *hash*, secara umum gambaran sistem ini adalah pengirim pesan membuat *pair key*, dan pengirim akan dapat mengirim pesan kepada penerima dengan menggunakan *private key* yang sudah dibuat sebelumnya sementara *public key* di simpan di dalam *directory* dan *hash* untuk mengubah file maupun *digital signature* ke dalam bentuk *digest* yang nanti akan dikirim, karena penerima pesan ingin memastikan bahwa pesan yang diterima adalah aseli dari pengirim yang tepat, maka harus dilakukan validasi terhadap *digital signature* tersebut dan memastikan isi dari pesan atau dokumen itu tidak ada yang merubah atau menggantinya ketika diperjalanan menuju ke penerima dengan menggunakan *public key* yang sudah tersimpan di *directory*.

Sebelum melakukan pengiriman pesan kepada penerima, terlebih dahulu harus membuat pasangan kunci *public key* dan juga *private key* yang nanti akan di pakai untuk memvalidasi pesan beserta *digital signature* yang akan dikirim kepada penerima.

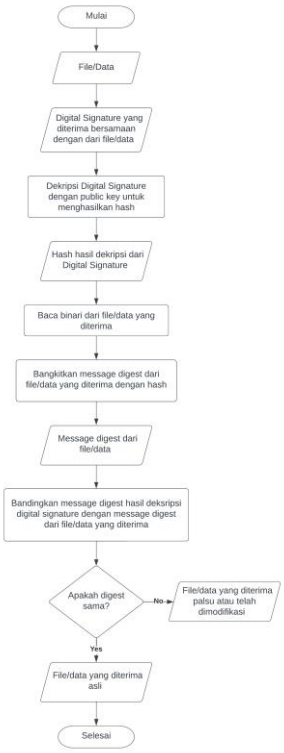
Proses yang dilakukan dalam membentuk *digital signature* pada data dapat dilihat pada flowchart yang ditujukan gambar dibawah.



Gambar Proses *Digital Signature File*

Sedangkan proses dalam melakukan pengujian keaslian data dapat dilihat pada flowchart yang ditujukan pada gambar dibawah ini.

ISSN 2337-6805



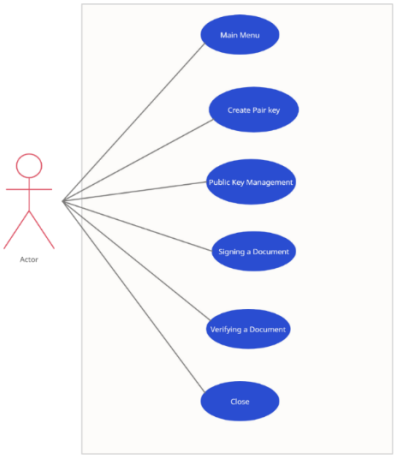
Gambar Proses Pengujian keaslian *File*

Desain Perancang Sistem

Perancangan aplikasi dilakukan dengan membuat *Unified Modeling Language* (*UML*) yang merupakan standar dalam pembangunan aplikasi berbasis objek. Berikut ini adalah hasil dari perancangan aplikasi berupa 3 Model *UML* yang digunakan:

Use Case Diagram

Diagram *use case* merupakan gambaran dari interaksi yang terjadi antara komponen-komponen suatu sistem yang dibangun. Diagram *use case* juga dapat mendokumentasikan persyaratan sistem dengan baik.

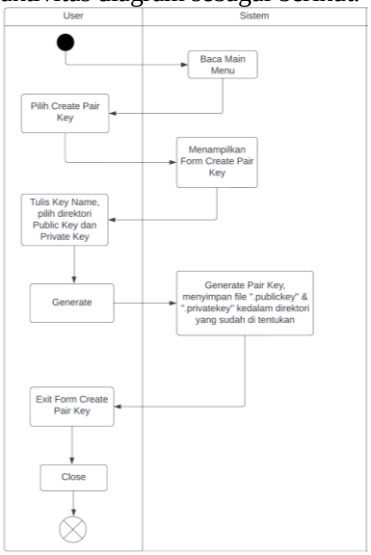


Gambar *Use Case Diagram*

Terlihat pada gambar 5 dimana dalam proses *use case diagram, actor* dapat melakukan beberapa kegiatan, yang pertama yaitu *actor* dapat *form main menu* yang menampilkan proses dari pembuatan *digital signature* sampai verifikasi dari aplikasi yang akan dibuat. Kemudian terdapat *create pair key* untuk proses pembangkitan pasangan kunci dari *public key* dan juga *private key*, selain itu ada *public key management* untuk menyimpan *public key* kedalam *directory*, setelah itu masuk kedalam proses *signing a document* yang akan dilakukan oleh *actor* dimana ada proses *signing* pasti ada proses untuk validasi nya atau *verifying a document* untuk memastikan keaslian dari file yang dikirim dari pengirim, dan terakhir yaitu *close* untuk keluar dari aplikasi.

Activity Diagram

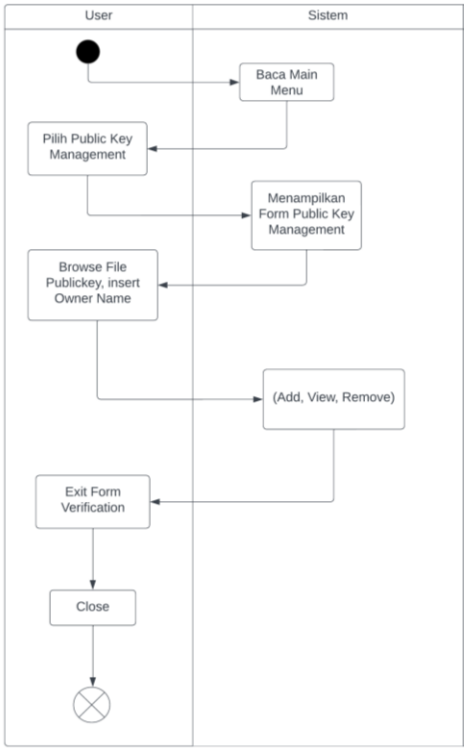
Activity diagram memodelkan *workflow* proses dan urutan aktivitas dalam sebuah proses. Untuk menggambarkan proses kerja dari sistem yang sedang berjalan maka dibuatlah aktivitas diagram sebagai berikut.



Gambar *Activity Diagram* Proses Pembuatan *Pair Key*

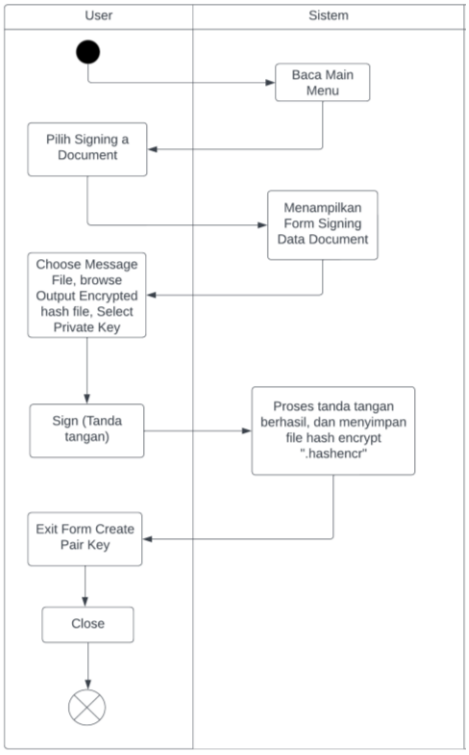
Seperti yang di tunjukan pada gambar *Activity Diagram* Proses Pembuatan *Pair Ke*, hal pertama yang dilakukan untuk membuat dan memverifikasi *digital signature* adalah membuat pasangan kunci (*pair key*) untuk dapat memastikan keaslian atau integritas dari data *digtial signature* yang akan dibuat.

ISSN 2337-6805



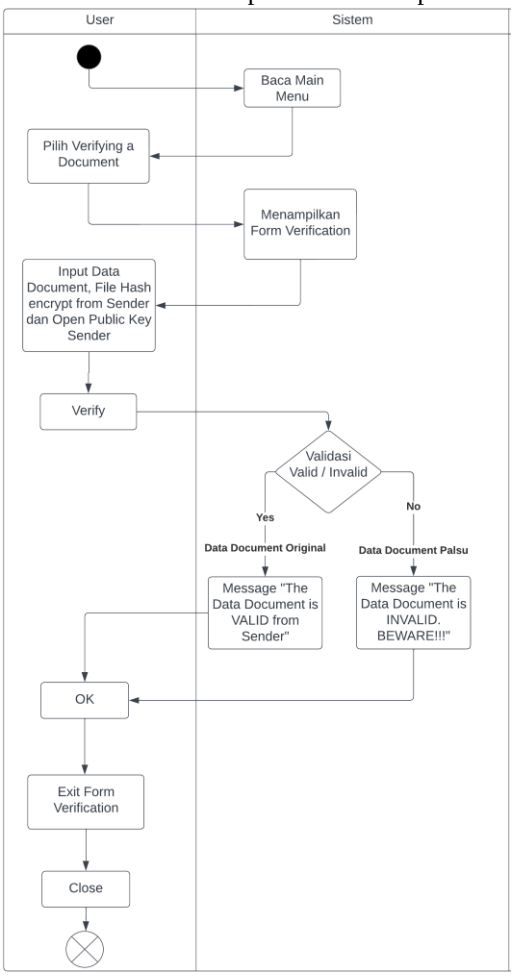
Gambar *Activity Diagram* Proses Penyimpanan Direktori PublicKey

Selanjutnya terlihat pada gambar Activity Diagram Proses Penyimpanan Direktori PublicKey yaitu proses penyimpanan direktori *public key* dengan memasukan tempat penyimpanan atau direktori *public key* beserta nama pemiliknya kedalam database agar ketika program aplikasi atau *laptop* dimatikan data *public key* masih tersimpan.



Gambar *Activity Diagram Signing Document*

Setelah mempunyai pasangan kunci (pair key) maka dibuatlah *digital signature* yang ditujukan pada gambar 8 yaitu dengan meng-enkripsi pesan menggunakan *hash* dan juga *private key* yang sudah dibuat sebelumnya, kemudian pesan tersebut di *hash* untuk keamanan yang lebih tinggi karena *output* dari algoritma *hash* sendiri yaitu berbentuk *digest* dan sulit untuk di dekripsi atau di manipulatif.



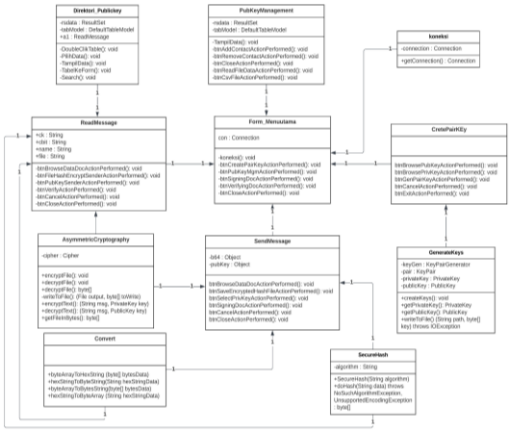
Gambar *Activity Diagram* Verifikasi *Digital Signature*

Kemudian pada gambar Activity Diagram Verifikasi Digital Signature menunjukan proses verifikasi dokumen, dimana penerima dokumen akan menerima pesan yang ter-integritas keamanan dan keaslian dari isi pesan tersebut dengan membuktikannya atau memverifikasinya dengan menggunakan *publickey* yang tersimpan di *directory* dan juga dengan file *hashencr* dari pengirim dokumen, ketika dokumen memang asli dari pengirim maka program akan memvalidasi bahwa dokumen atau pesan tersebut *valid*.

ISSN 2337-6805

Class Diagram

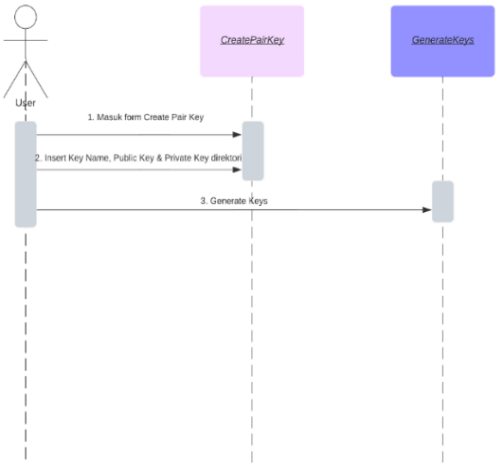
Class diagram menggambarkan struktur statis dari kelas dalam sistem dan menggambarkan atribut, operasi dan hubungan antara kelas. Selama tahap desain, *class diagram* berperan dalam menangkap struktur dari semua kelas yang membentuk arsitektur sistem yang dibuat. Berikut ini merupakan *class diagram* dari aplikasi *Digital Signature* pada gambar dibawah ini



Gambar *Class Diagram*

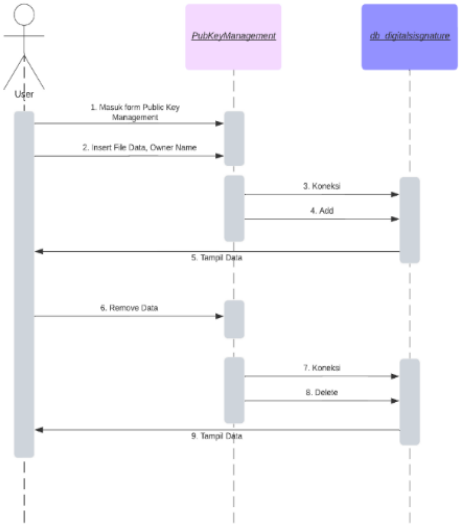
Sequence Diagram

Sequence diagram merupakan diagram yang menggambarkan pola hubungan diantara sekumpulan objek yang saling mempengaruhi menurut urutan waktu. Sebuah objek berinteraksi dengan objek lain melalui pengiriman pesan. Berikut ini *sequence diagram* pada *create pair key* pada aplikasi *digital signature* yang ditujukan pada gambar dibawah ini :



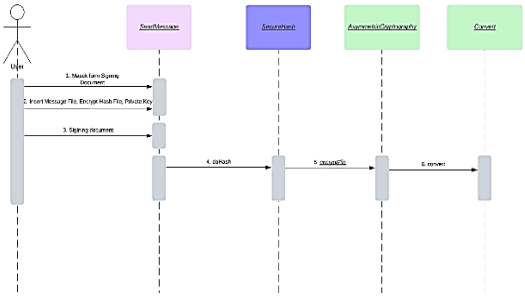
Gambar *Sequence Diagram Create Pair Key*

Berikut ini adalah *sequence diagram* dari *public key management* yang di tujukan pada Gambar dibawah ini :



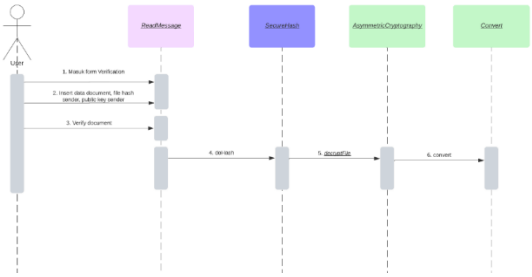
Gambar *Sequence Diagram Public Key Management*

Berikut ini adalah *sequence diagram* dari proses *signing a document* yang di tujukan pada Gambar dibawah ini :



Gambar *Sequence Diagram Signing Document*

Berikut ini adalah *sequence diagram* dari proses verifikasi *digital signature* yang di tujukan pada Gambar dibawah ini :



Gambar *Sequence Diagram Verification*

Deployment Diagram

Deployment diagram menggambarkan proses-proses yang berbeda pada sistem yang berjalan dan bagaimana relasi di dalamnya. *Deployment* diagram dapat dianggap sebagai ujung spektrum dari kasus penggunaan, menggambarkan bentuk fisik dari sistem yang bertentangan dengan gambar konseptual dari pengguna dan perangkat berinteraksi dengan sistem. Berikut ini merupakan *deployment*

ISSN 2337-6805

diagram pada aplikasi *digital signature* yang ditujukan pada gambar dibawah ini :

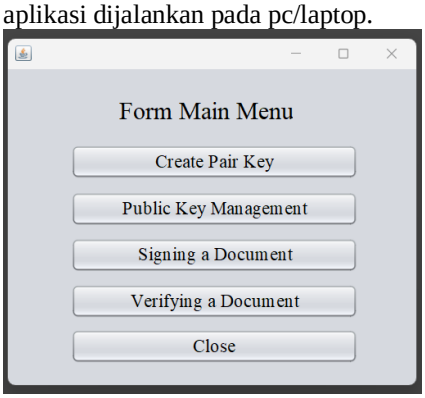
Gambar *Deployment Diagram*

3. HASIL DAN PEMBAHASAN

Hasil Pengujian

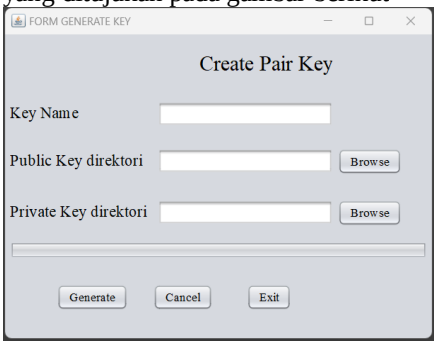
Pada pengujian aplikasi *Digital Signature* dengan algoritma RSA dan hash menggunakan metode *blackbox testing*. Dari hasil pengujian, untuk melakukan proses enkripsi dan dekripsi pesan menggunakan satu buah perangkat. Pada perangkat tersebut berfungsi sebagai *user* yang menggunakan autentifikasi *digital signature* tersebut. Berikut ini merupakan hasil pengujian aplikasi *digital signature*:

Main menu akan muncul pertama kali ketika aplikasi dijalankan pada pc/laptop.



Gambar Tampilan *main menu*

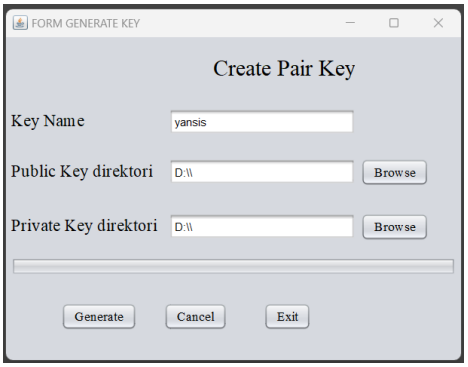
Selanjutnya kita memilih membuat *pair key* terlebih dahulu untuk langkah pertama seperti yang ditujukan pada gambar berikut



Gambar. Tampilan *Form Create Pair Key*

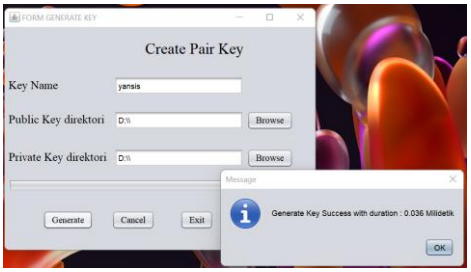
Pada gambar Tampilan Form Create Pair Key merupakan tampilan awal dari *form create pair key*, dimana terdapat *text field* untuk mengisi *key name*, *public key* direktori, dan juga *private key* direktori dan ada *button* untuk *browse* kemudian *generate*, *cancel*, dan juga *exit*.

Berikut di bawah ini adalah *form create pair key* yang sudah terisi ditujukan pada Gambar berikut :



Gambar Tampilan *Form Create Pair Key* diisi

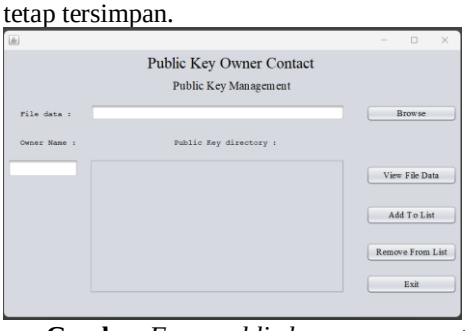
Pada gambar Tampilan Form Create Pair Key diisi ini merupakan tampilan dari form yang sudah di isi oleh user.



Gambar Tampilan *alert success generate form Create pair key*

Pada gambar Tampilan alert success generate form Create pair key user sudah berhasil melakukan *generate pair key* dengan duration 0.036 Milidetik dan hasil dari *pair key* tersimpan kedalam direktori D:\\

Setelah membuat pair key, maka sistem akan masuk ke halaman menu utama dan proses selanjutnya kita memilih public key management untuk memasukan direktori public key kedalam database, agar ketika aplikasi atau laptop dimatikan public key akan tetap tersimpan.



Gambar *Form public key management*

ISSN 2337-6805

Pada gambar Form public key management ini merupakan tampilan awal dari *form public key management*, dimana terdapat dua *text field* kosong untuk mengisi file data dan juga *owner name* atau pemilik dari *public key*. Selain itu juga terdapat beberapa *button* yang memiliki fungsinya masing-masing yaitu *button view file data, add to list, remove from list* dan juga *exit*.
Berikut adalah tampilan dari *form public key management* yang sudah terisi ditujukan pada Gambar berikut

Gambar Proses isi data *form public key management*

Pada gambar Proses isi data form public key management terlihat bahwa *text field* yang kosong sudah terisi untuk melakukan proses simpan data kedalam *database*.

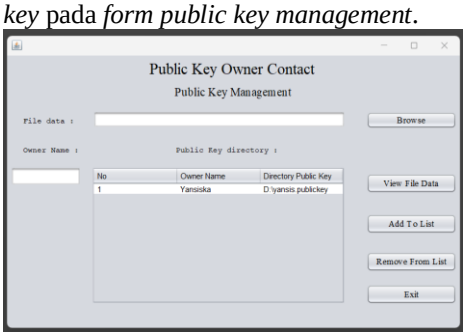
Gambar Proses *save directory public key*

Tampilan ketika proses simpan data berhasil terlihat pada gambar Proses *save directory public key* dimana muncul *alert* juga ketika proses berhasil dijalankan.

Gambar Proses setelah simpan data

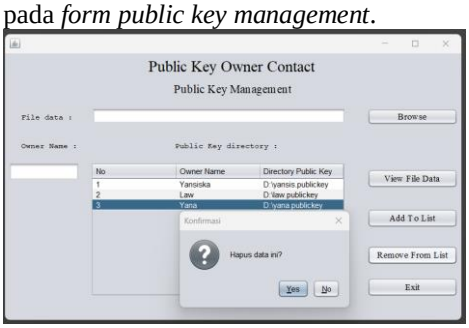
Pada gambar Proses setelah simpan data menunjukkan ketika proses simpan data berhasil, dan menekan "ok" pada *alert* maka muncul tabel dan menampilkan data yang baru saja sudah disimpan kedalam *database*.

Jika ingin menampilkan file *directory public key* pada *form public key management*.



Gambar Proses *view* direktori *public key*

Pada gambar Proses view direktori public key terlihat proses untuk menampilkan data, dengan menekan *button view file data*.
Jika ingin menghapus *file directory public key* pada *form public key management*.



Gambar Proses hapus data

Pada gambar Proses hapus data terlihat proses untuk menghapus data, dengan menekan tabel dari data yang ingin dihapus kemudian klik *button remove from list* dan hasilnya akan terlihat pada gambar dibawah ini.

Gambar Proses Setelah hapus data

Selanjutnya yaitu mengisi *form signing data document* untuk membuat dan mengirim pesan.

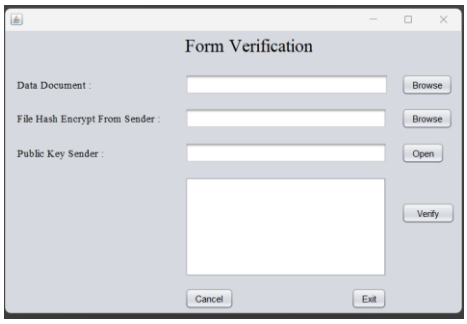
Gambar *Form signing data document*

ISSN 2337-6805

Pada gambar Form public key management ini merupakan penampilan awal dari *form signing document* ketika dijalankan.

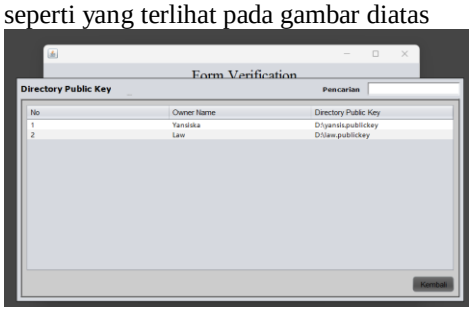
Gambar Proses isi data *form signing document*

Gambar diatas menunjukan ketika *form signing data document* terisi dan menekan tombol sign maka akan muncul *alert* yaitu "Dokumen telah di tandatangani. Lihat file *.hashencr".
Setelah mengirim pesan kemudian penerima ingin memastikan bahwa pesan tersebut berasal dari orang yang tepat maka dilakukan lah proses verifikasi dengan masuk ke form verification.



Gambar *form verifikasi document*

Tampilan awal pada form verification akan seperti yang terlihat pada gambar diatas



Gambar. Proses ambil data *public key* direktori

Pada gambar diatas ini adalah proses untuk mengambil data *public key* milik pengirim yang mana sudah di simpan di dalam direktori *public key* ketika *button open* di tekan maka akan muncul tampilan data *public key*.

Gambar Proses isi data *form verification* berhasil

Gambar diatas menunjukan ketika proses verifikasi berhasil maka akan muncul *alert* "The Data Document is VALID from the sender"
Jika isi pesan sudah di ganti atau ada yang mengubah ketika di perjalanan menuju ke penerima maka akan seperti yang ditujukan gambar di bawah ini.

Gambar Proses isi data *form verification* tidak berhasil

Terlihat pada *output textarea* di *verify signature data document* dan *signature form hash* tidak sama *output* keluarannya yaitu untuk *signature the data document* adalah "4197d561804e35e17f6a46e6717a8246" sementara untuk *output* dari *signature from hash* adalah "4f4c4486549f08589f12dce28dd7a445" karena pesan dari isi dokumen sendiri sudah berubah.
Jika file *hash* diganti menggunakan file *hash* dari pengirim lain maka akan seperti gambar dibawah ini.

Gambar proses verifikasi ketika file *hash* diganti

Terlihat pada *output* textarea di *verify signature data document* dan *signature form hash encrypted* tidak sama *output* keluarannya yaitu untuk *signature the data document* adalah

ISSN 2337-6805

"4f4c4486549f08589f12dce28dd7a445" sementara untuk output dari *signature from hash encrypted* adalah "fdc613a5c6710d8ecc25325e95f45b86" karena file *hash encrypted* sudah diganti dengan yang lain.
Jika melakukan verifikasi menggunakan public key bukan dari pengirim data maka tidak akan valid seperti gambar dibawah.

Gambar Proses verifikasi ketika menggunakan *public key* pengirim lain

Analisa Kerja Sistem
Analisa hasil pengujian diperoleh setelah melakukan pengujian pada aplikasi yang sudah dibuat. Hal ini berdasarkan pada metode *black box* yaitu melakukan pengujian berdasarkan apa yang dilihat, hanya fokus terhadap fungsionalitas dan *output*. Hasil pengujian yang didapat bahwa aplikasi *digital signature* dapat digunakan pada *laptop* atau *PC* dengan sistem operasi berbasis *windows*.
Pada analisa kerja dari aplikasi *digital signature* menggunakan algoritma *RSA* dan *Hash* ini merupakan tahapan penguraian, penelahan, dan pendalaman terhadap sistem yang bekerja secara normal maupun kritis. Pada bagian sebelumnya telah dijelaskan prosedur pengujian kerja sistem secara umum agar mempermudah dalam menganalisa kerja dari sistem ini.
Selanjutnya, untuk menjalankan aplikasi *digital signature* yang pertama kali akan muncul adalah tampilan *main menu* untuk melanjutkan ke tahap selanjutnya. Jika sudah masuk ke *main menu* akan muncul beberapa pilihan *icon menu* dengan fungsinya masing-masing, diantaranya:

Create Pair Key
Pada *main menu* terdapat *button* untuk menuju ke *form create pair key*, setelah masuk kedalam *form create pair key* tersedia *field-field* yang harus di isi dan ada *button generate* untuk membuat pasangan kunci *public* dan kunci *private*, kemudian *button cancel* untuk *cancel* pembuatan *pair key*, dan *button exit* untuk keluar dari *form create pair key* menuju ke *main menu*.

Public Key Management
Pada *main menu* terdapat *button* untuk menuju ke *form public key management* dimana di dalam nya ada *button* untuk menampilkan data, menyimpan data, menghapus data dan ada satu *table* untuk menampilkan *output* dari *button-button*. Untuk menyimpan data juga ada 2 *textfield* untuk menginput data dan *exit* untuk keluar dari *form public key management*.

Signing a Document
Pada *main menu* terdapat *button* untuk menuju ke *form signing document* dimana di dalamnya terdapat *textfield* untuk mengisi *file/message*, menyimpan *output hash file* enkripsi dari *file/message*, dan *private key*. Untuk mengirim dan menandatangani pesan ada *button sign*, dan *exit* untuk keluar dari *form signing document*.

Verifying a Document
Pada *main menu* terdapat *button* untuk menuju ke *form verification* dimana di dalamnya ada proses untuk memverifikasi *file/message* yang sudah dikirim, ada *textfield file/message* yang akan di verifikasi kemudian sertakan *hash file* hasil enkripsi dari pengirim dan pilih *public key* pengirim setelah itu klik *button* untuk memverifikasi dan akan memunculkan *alert valid* atau *invalid*.
Close
Pada *main menu* terdapat *button close* untuk menutup aplikasi.

4. SIMPULAN

Digital Signature adalah teknologi yang digunakan untuk melakukan pertukaran pesan atau dokumen dengan aman dan terbukti keasliannya, sehingga tidak akan ada yang bisa memanipulasi pesan atau dokumen yang dikirim.
Hasil pengimplementasian *digital signature* dengan menggunakan *NetBeans IDE* adalah sebagai berikut.
Aplikasi yang telah dibuat dengan menggunakan NetBeans dan menggunakan bahasa pemrograman java dapat dijadikan indikator keaslian sebuah dokumen.
Penerima (*receiver*) dapat mengetahui apakah dokumen yang dikirimkan merupakan dokumen asli atau dokumen yang telah diubah.

ISSN 2337-6805

5. DAFTAR PUSTAKA

[1] Azdy, R. A. (2016). Tanda Tangan Digital Menggunakan Algoritme Keccak dan RSA. *JNTETI, 5(3)*, 184-191.

[2] Hidayat, A., & Sidik, I. M. Z. (2019). Digital Signature Menggunakan Algoritma RSA Pada Dokumen PDF. Universitas Siliwangi. Tasikmalaya

[3] Ihwani, M. (2016). Model Keamanan Informasi Berbasis Digital Signature Dengan Algoritma RSA. *CESS (Journal Of Computer Engineering System And Science), 1(1),* 15-20.

[4] Mahdiana, D. (2011). Analisa dan rancangan sistem informasi pengadaan barang dengan metodologi berorientasi obyek: studi kasus PT. Liga Indonesia. *Jurnal TELEMATIKA MKOM, 3(2),* 36-43.

[5] Pradipta, R. A., Wintoro, P. B., Budiyanto, D. (2022). Perancangan Pemodelan Basis Data Sistem Informasi Secara Konseptual Dan Logikal. *JITET (Jurnal Informatika dan Teknik Elektro Terapan), 10(2),* 127-132.

[6] Putri, Y. A., & Manullang, E. V. (2018). Implementasi Kriptografi Digital Signature Menggunakan Secure Hash Algorithm (SHA-1). *Jurnal Teknologi Informasi, 6(1)*, 1-8.

[7] Sitinjak, D. D. J. T., Maman., Suwita, J. (2020). Analisa Dan Perancangan Sistem Informasi Administrasi Kursus Bahasa Inggris Pada Intensive English Course Di Ciledug Tangerang. *JURNAL IPSIKOM, 8(1),* 1-19.

[8] Suhandinata, S., Rizal, R. A., Wijaya, D. O., Warren, P., & Srinjiwi. (2019). Analisis Performa Kriptografi Hybrid Algoritma Blowfish Dan Algoritma RSA. *JURTEKSI (Jurnal Teknologi dan Sistem Informasi), 6(1),* 1-10.

[9] Suharya, Y., & Widia, H. (2020). Implementasi Digital Signature Menggunakan Algoritma Kriptografi RSA Untuk Pengamanan Data Di Smk Wirakarya 1 Ciparay. *Jurnal Informatika – COMPUTING 7(1),* 20-29.

[10] Warasart, M., & Kuacharoen, P. (2012). Paper-based Document Authentication using Digital Signature and QR Code. *4TH International Conference on Computer Engineering and Technology (ICCET),* 1-5.

[11] Yusfrizal. (2019). Rancang Bangun Aplikasi Kriptografi Pada Teks Menggunakan Metode Reverse Chiper Dan Rsa Berbasis Android. *Jurnal Teknik Informatika Kaputama (JTIK) 3(2),* 29-37.